

Portable Program Review

Vol. 1, No. 3

September, 1985

THINK ABOUT IT A WHILE

So-called thought processors and outline formatters are among the latest computing fads. Outlines are still a useful tool for brainstorming and drafting documents, and thought processors reflect that process. THINK.IT is an outline processor for the Model 100.

THINK.IT immediately asks for a filename. If beginning a new outline, press Enter. The program prompts for a filename to save existing work in when exiting. Use this name at the opening screen in future sessions to reedit this outline.

Initially the outline consists of only one item. Additional entries can be added at the same level as the topic the cursor is resting on by pressing Enter. Subheadings are entered under the cursor with the Tab key. Outline entries can be up to 250 characters long. Entries scroll, making room for further text. The shift, control and arrow keys allow easy TEXT-like editing of entries. Pressing Enter exits the edit mode.

The vertical arrows navigate the complete outline. The current item will be highlighted in reverse video and indented as appropriate. The shift and control keys work here as well. Continued on Page

2

THE COOPERATIVE LAPTOP COMPUTER

User-friendly is a catch-phrase in the computer industry, with good reason. Execution speed, memory efficiency and comprehensive output options are plusses -- but the acid test comes at the keyboard. If the program's not intuitively easy to run, then it simply won't be used.

Appearances can be deceiving, but first impressions are all-important, especially in the software business. A complex escape code-driven word processor will sit idle -- while a more limited but menu-based package is hailed as revolutionary.

Here are some ideas for making your own revolutionary software, or testing the friendliness of commercial packages.

If you have suggestions for making software friendly to use -- or know of techniques to avoid -- please share them with other Tandy owners. Continued on Page

6

THINK ABOUT IT

THINK.IT's author, Larry Groebe, is a programmer working for the Fort Worth Star-Telegram's videotext project, STARTEXT. He's also worked for Tandy's software engineering department. -Ed.

+AMAZING SAMPLE OUTLINE

- +Why THINK.IT is a nice program.
 - Fairly small size.
 - Runs faster than competition.
 - Cheaper, too!
- +Why Larry Groebe is a nice guy.
 - He worked months on the program.
- +Limitations of THINK.IT
 - +250 chars each idea.
 - No paragraphs, either.
 - 200 ideas maximum.

Lower levels of the outline can be hidden by pressing the left arrow. The right cursor causes them to reappear. Shift-left and Shift-right will collapse the outline to its most major points and expand it to cover all entries, respectively.

The various function keys perform the operations they indicate: FIND, VIEW, PRINT, CUT, COPY, MENU.

FIND searches for words or phrases and moves the cursor to that entry in the outline.

VIEW changes the view of the outline. VIEW-1 displays all items on screen from the top level to the lowest one selected by the left-and-right arrows. VIEW-2 displays only items on the same level, and hides those both below and above. To see the subtopics of an outline entry in VIEW-2, place the cursor on the desired entry and press the right arrow. That entry will vanish and its subentries only will appear. To return to the parent entry, press the left arrow.

PRINT sends the outline to a printer or RAM file for further editing with TEXT. A print width of zero is recommended when outputting to RAM.

CUT and COPY excise or copy entire collections of thoughts. The entry at the cursor and its sub-entries will be placed in a buffer. PASTE recalls these entries. To cut a single entry, but not its sub-entries, use

the Shift-DEL key combination. Once an entry has been PASTED, it is removed from the buffer and cannot be pasted again.

Pressing F8 exits THINK.IT. The file will be saved at this time. Pressing SHIFT-F8 will exit without saving the outline.

Pressing the plus (+) key moves all items and sub-items under the cursor up one level of importance. Conversely, minus (-) will drop items down a level in the outline.

Key Summary

CURSOR KEYS

- | | |
|---------------|---|
| * Up, Down | --Moves between outline items. |
| * Shift | --Moves screenful of items. |
| * Control | --Moves to first or last item. |
| * Left, Right | --Collapses or expands outline one level. |
| * Shifted | --Collapses or expands outline to max or min. |
| * Controlled | --Scrolls long entry under cursor to reveal more. |

ADDING OUTLINE ITEMS

- | | |
|---------|---|
| * Enter | --Begins editing new thought after current, same level. |
| * Tab | --Begins editing new thought after current, next level. |
| * Space | --Allows editing of existing thought. |

IN EDITING MODE

- | | |
|----------|---|
| * Escape | --Erases text from an entry. |
| * PASTE | --Pastes any cut or copied entries beginning at current location. |
| * CODE-c | --Converts numeric expression to a number. For example, SIN(1)/3 gets replaced by the number .2804. |

The left and right arrows, shifts and CTRL, DEL, BKSP work like regular TEXT during editing. The vertical arrows and Enter quits EDIT.

OTHER FUNCTIONS

* + key	--Raises cursor item and all following items up a level.
* - key	--Lowers cursor item and all that follow down a level.
* F1	--FIND. Enter text (ESC clears existing) and search for it.
* F2	--Toggles between VIEW-1 and VIEW-2.
* F4	--Brings up PRINT routine.
* F5	--COPYs idea and sub-ideas to paste buffer.
* F6	--CUTs idea and sub-ideas from outline, stores in buffer.
* DEL	--Cuts only single idea from outline. Any sub-ideas are promoted one level.
* F8	--Exits to MENU & saves outline
* SHIFT-F8	--Exits without saving outline.

Listing for THINK.IT, the outline processor.

```

0 REM **THINK.IT (C)1985 Larry Groebe**
  Permission is granted to distribute if
  line 0 is included. If you like this,
  and want the complete package, send $20
  to Larry Groebe, 6001 Skillman #362,
  Dallas, TX 75231
1 MAXFILES=1: CLEAR 0, HIMEM: CLEAR FRE(0)/2
: DEFSTR A: DEFINT B-Z: X=0: Y=0: L=1: M=1
: V=0: U=0: T=0: DIM P(200), N(200), A(200)
: E$=CHR$(27): R1$=E$+"p": R2$=E$+"q"
: ER$=E$+"K": IN$=E$+"L": L7$=E$+"T"

```

```

: U7$=E$+"U": P$="+": M$="-": AA=R1$
+SPACE$(36)
2 I=2: MT=0: D$="LPT:": ES=1: LS=1: TI=4
: NU$="n": W0=75: S=6
3 AR=CHR$(241)+CHR$(154): CC$=
"?>@A#(6$349</+B=*"+CHR$(161): FOR Z=1 TO
18: MID$(CC$, Z, 1)=CHR$(ASC(MID$(CC$, Z))
-34): NEXT: SCREEN , 0: CLS: AB=">THINK.IT>
(C)1985 L.Groebe CV.1": ON ERROR GOTO
155
4 ON KEY GOSUB 75, 146, 1, 81, 59, 60, 1, 142
: GOSUB 137: CLS: GOSUB 13: GOTO 15
5 PRINT @0, R2$:; FOR T=0 TO S
6 IF ASC(A(U))>M THEN U=N(U): GOTO 6
7 N=ASC(A(U)): IF N<LT THEN 12
8 IF A(U)=A(X2) THEN Y0=T
9 Z=I*N-I: B$=M$: IF ASC(A(N(U)))>N THEN
B$=P$
10 PRINT @T*40, SPACE$(Z) B$ MID$(A(U),
2, 36-Z) ER$;: IF POS(0)=37 THEN PRINT AR;
11 U=N(U)
12 NEXT: PRINT E$"J";: RETURN
13 PRINT U7$;: PRINT @280, R1$"Find View
Prnt Copy Cut Menu";
: LINE(239, 63)-(233, 56), 1, BF: PRINT
@289, CHR$(49-MT) E$ "H" L7$: RETURN
14 IF MT THEN M=1
15 LT=1: U=1: GOSUB 5: X=1: Y=0: Q=1
16 GOSUB 132
17 KEY ON: A=INKEY$: IF A="" THEN 17
18 KEY OFF: IF A=P$ THEN GOSUB 67: GOTO 16
19 IF A=M$ THEN GOSUB 74: GOTO 16
20 B=INSTR(CC$, A): IF B=0 THEN 17
21 ON B GOSUB 47, 46, 39, 43, 23, 24, 51, 56, 27,
27, 14, 55, 32, 29, 49, 28, 28, 63
22 GOTO 16
23 LT=1: M=1: GOTO 48
24 IF NOT MT THEN M=LU: GOTO 48: ELSE BEEP
: RETURN
27 ZB=ZA: ZA=ZA+(37-I*L)*SGN(B-9.5): IF ZA>=0
AND ZA<LEN(A(X)) THEN GOSUB 132: PRINT
ER$; ELSE ZA=ZB
28 RETURN
29 IF L>8 THEN BEEP: RETURN
30 IF MT THEN Y=0: L=L+1: LT=L: M=L: U=N(X)
: GOSUB 5: PRINT @0, IN$ ELSE GOSUB 134
: PRINT @40*Y+I*L-I, "+": L=L+1
31 Q=Q+1: Y0=1: IF L>LU THEN LU=L: M=L: GOTO 33
ELSE 33
32 GOSUB 134: U=X: GOSUB 120: X=V: Q=Q+QC: IF
Y+Y0>S THEN U=X: Y1=Y: GOSUB 127: U=V: GOSUB
5: Y0=1

```



```

33 Y=Y+Y0:PRINT @40*Y-1,"":PRINT IN$;:IF
Y>S THEN Y=S
34 A=" ":H=I*(L-1)+1:G=38-H:H=40*Y+H:E=1
:F=1
35 GOSUB 102:IF A=CHR$(0) THEN 94
36 A(IT)=CHR$(L)+LEFT$(A,LEN(A)-1)
:P(N(X))=IT:N(IT)=N(X):P(IT)=X:N(X)=IT
:X=IT:IT=IT+1:QT=QT+1
37 U=X:GOSUB 125:IF U<>X THEN U=X:Y=Y-1
:Y1=Y:GOSUB 127:U=V:GOSUB 5
38 RETURN
39 IF ASC(A(P(X)))<LT THEN RETURN
40 GOSUB 134:FOR Q=Q-1 TO 1 STEP -1:X=P(X)
:IF ASC(A(X))>M THEN NEXT
41 IF Y>0 THEN Y=Y-1 ELSE PRINT @0,IN$
42 RETURN
43 OQ=Q:OX=X:GOSUB 134:FOR Q=Q+1 TO 200
:X=N(X):IF ASC(A(X))>M THEN NEXT
44 IF ASC(A(X))<LT THEN X=OX:Q=OQ:RETURN
45 IF Y<S THEN Y=Y+1:RETURN ELSE PRINT
:RETURN
46 IF M=LU THEN RETURN ELSE M=M+1:IF NOT MT
THEN 48 ELSE IF ASC(A(N(X)))=M THEN
X=N(X):Q=Q+1:LT=M:GOTO 48 ELSE M=M-1
:RETURN
47 IF M=1 THEN RETURN ELSE M=M-1:IF MT THEN
LT=M
48 GOSUB 125:U=X:Y1=Y:GOSUB 127:U=V:X2=X
:GOSUB 5:Y=Y0:RETURN
49 E=ZA+1:F=E:H=I*L-1:G=39-H:H=40*Y+H
:A=MID$(A(X),2)+" ":PRINT @0,R2$:GOSUB
102:A(X)=LEFT$(A(X),1)+LEFT$(A,LEN(A)-1)
:ZA=0:RETURN
50 GOSUB 125:U=X:GOSUB 127:RETURN
51 GOSUB 134:Y=Y-((S+1)*(Y=0)):FOR T=1 TO Y
52 M1=ASC(A(P(X))):IF M1>=LT THEN X=P(X)
:Q=Q-1:IF M1>M THEN 52
53 NEXT:U=X:IF Y>S THEN GOSUB 5
54 Y=0:RETURN
55 X=P(1):X=P(X):Y=S:L=ASC(A(X)):Q=QT-1
:GOTO 48
56 GOSUB 134:Y=Y+(S+1)*(Y=S):FOR T=Y TO 5
57 M1=ASC(A(N(X))):IF M1>=LT THEN X=N(X)
:Q=Q+1:IF M1>M THEN 57
58 NEXT:L=ASC(A(X)):Y=S:GOTO 48
59 DZ=-1
60 CZ=0:CX=X:U=X:GOSUB 120:CY=V:CQ=QC:IF DZ
THEN SOUND 500,5:CZ=-1:DZ=0:RETURN
61 N(P(CX))=N(CY):P(N(CY))=P(CX):X=P(CX)
:P(CX)=-1:N(CY)=-1:IF MT THEN IF
ASC(A(X))<=L THEN LT=ASC(A(X)):M=LT
62 L=ASC(A(X)):Q=Q-1:QT=QT-QC:U=X:GOSUB 48

```

```

:GOSUB 132:RETURN
63 QT=QT-1:Q=Q-1:CX=X:CY=X:U=X:GOSUB 120
64 N(P(CX))=N(CX):P(N(CX))=P(CX):X=P(CX)
:P(CX)=-1:U=CX:IF MT THEN IF
ASC(A(X))<=L THEN LT=ASC(A(X)):M=LT
65 IF U=V THEN L=ASC(A(X)):GOTO 48
66 U=N(U):L=ASC(A(U))-1:A(U)=CHR$(L)+
MID$(A(U),2):GOTO 65
67 U=X:Z=-1:N=L:IF N=1 THEN 17
68 MID$(A(U),1,1)=CHR$(N+Z)
69 IF N=LU THEN LU=LU+1:M=M+1
70 U=N(U):N=ASC(A(U)):IF N=>L THEN 68
71 U=X:Y1=Y:L=ASC(A(X)):IF MT THEN LT=L:M=L
72 IF L>M THEN GOSUB 125
73 GOSUB 127:U=V:X2=X:GOSUB 5:Y=Y0:RETURN
74 U=X:Z=1:N=L:IF ASC(A(P(X)))>=L THEN 68
ELSE BEEP:RETURN
75 PRINT @0,U7$:PRINT @280,R1$ "Find?"
SPACE$(34);:A=AZ+" ":E=1:F=1:G=31:H=287
:GOSUB 102:A=LEFT$(A,LEN(A)-1):IF A=AZ
THEN X=N(X)
76 AZ=A
77 FOR Q=Q TO 199:IF INSTR(A(X),AZ)>0 THEN
79 ELSE IF ASC(A(X))>0 THEN X=N(X):NEXT
78 GOSUB 13:GOTO 14
79 L=ASC(A(X)):IF L>M THEN M=L:IF MT THEN
LT=L
80 GOSUB 13:GOSUB 48:GOTO 132
81 U=X:CLS:PRINT R2$"Print to "D$;:INPUT D$
:PRINT "Indent" TI "spaces/level";
:INPUT TI:PRINT "Line Spacing of "LS;
:INPUT LS:PRINT "Lowest level for
separation is"ES;:INPUT ES
82 N$="":FOR N=64 TO 74:N$=N$+CHR$(1):NEXT
:PRINT "Numbering ("NU$)";:INPUT NU$
:PRINT "Line Width of"W0;:INPUT W0:IF
W=00 THEN W0=999
83 CLOSE:OPEN D$ FOR OUTPUT AS 1:CLS:PRINT
@50,"Writing to "D$
84 N=ASC(A(U)):IF N>M THEN U=N(U):GOTO 84
85 IF N<L THEN U=X:GOSUB 48:GOTO 132
86 NL=N-L+1:IF N>ASC(A(P(U))) THEN
MID$(N$,NL,1)=CHR$(1) ELSE MID$(N$,NL,1)
=CHR$(ASC(MID$(N$,NL))+1)
87 A="":IF NU$="y" THEN FOR Z=1 TO NL:B=
ASC(MID$(N$,Z)):A=A+RIGHT$(STR$(B),1-
(B>9))+".":NEXT:A=LEFT$(A,LEN(A)-1)+": "
88 A=A+MID$(A(U),2):TS=TI*NL:IF N<=ES OR
(N<ASC(A(P(U))) AND N<M) THEN PRINT #1,
89 IF LEN(A)<W0-TS THEN W1=LEN(A):GOTO 91
90 FOR W1=W0-TS+1 TO 1 STEP -1:IF MID$(
A,W1,1)<>" " THEN NEXT W1 ELSE W1=W1-1

```



```

91 PRINT #1,SPACE$(TS) LEFT$(A,W1);:IF LS>0
   THEN FOR Z=1 TO LS:PRINT #1,:NEXTZ
92 A=MID$(A,W1+1):IF A$>"" THEN 89
93 U=N(U):GOTO 84
94 IF CX=0 THEN BEEP:GOTO 34
95 IF NOT CZ THEN 98 ELSE X2=IT
96 A(IT)=A(CX):P(IT)=IT-1:N(IT)=IT+1
   :IT=IT+1:IF CX<>CY THEN CX=N(CX):GOTO 96
97 CY=IT-1:CX=X2:CZ=0
98 P(N(X))=CY:N(CY)=N(X):P(CX)=X:N(X)=CX
   :X=CX:N=ASC(A(CX)):U=X:CX=0:Z=L-N
   :QT=QT+CQ:IF Z=0 THEN 48
99 MID$(A(U),1,1)=CHR$(N+Z)
100 IF N=LU THEN LU=LU+1:M=LU
101 IF U=CY THEN 71 ELSE U=N(U):N=ASC(A(U))
   :GOTO 99
102 IF LEN(A)>250 THEN BEEP:A=LEFT$(A,250)
103 IF E>LEN(A) THEN E=E-1
104 IF E-F=>G THEN F=E-G+1
105 IF E<F THEN F=E:IF E<1 THEN E=1:F=1
106 PRINT @H,MID$(A,F,G);:PRINT @H+E-F,;
   :B$=INPUT$(1)
107 ON INSTR(CC$,B$)+1 GOTO 108,111,110,28,
   28,118,116,28,28,114,115,28,28,28,106,
   109,117,112,113
108 IF B$=CHR$(0) THEN A=B$:RETURN
109 IF B$<>CHR$(162) THEN C$=RIGHT$(A,1)
   :MID$(A,E)=B$+MID$(A,E):A=A+C$ ELSE
   GOSUB 150
110 E=E+1:GOTO 102
111 E=E-1:GOTO 105
112 IF E>1 THEN MID$(A,E-1)=MID$(A,E)
   :A=LEFT$(A,LEN(A)-1):GOTO 111 ELSE 106
113 IF E<LEN(A) THEN MID$(A,E)=MID$(A,E+1)
   :A=LEFT$(A,LEN(A)-1):GOTO 103 ELSE 106
114 E=1:F=1:GOTO 103
115 E=LEN(A):F=(E-G-1)*-(E>=G)+1:GOTO 103
116 E=INSTR(E+1,A," "):IF E=0 THEN B=18:GOTO
   115 ELSE 103
117 A=" ":PRINT @H,ER$;:GOTO 114
118 IF E>1 THEN FOR E=E-1 TO 1 STEP -1:IF
   MID$(A,E,1)<>" " THEN NEXT
119 GOTO 103
120 Y0=0
121 FOR QC=1 TO 199:T=ASC(A(N(U))):IF T<=M
   THEN Y0=Y0+1
122 IF T=0 THEN 124
123 IF T>L THEN U=N(U):NEXT
124 V=U:RETURN
125 FOR Q=Q TO 1 STEP -1:IF L>M THEN X=P(X)
   :L=ASC(A(X)):NEXT
126 RETURN

```

```

127 IF Y1=0 THEN 131 ELSE FOR Z=1 TO Y1
128 M1=ASC(A(P(U))):IF M1>=LT THEN U=P(U)
   ELSE 130
129 IF M1>M THEN 128
130 NEXT
131 V=U:RETURN
132 L=ASC(A(X)):Z=I*L-I:B$=M$:IF
   ASC(A(N(X)))>L THEN B$=P$
133 PRINT @40*Y+Z,R1$ B$
   MID$(A(X),2+ZA,37-Z);:RETURN
134 Z=I*L-I:B$=M$:IF ASC(A(N(X)))>L THEN
   B$=P$
135 PRINT @40*Y+Z,R2$ B$ MID$(A(X),2,36-Z);
   :IF POS(0)=37 THEN PRINT AR;
136 ZA=0:RETURN
137 PRINT @2,AA:PRINT @3,AB:PRINT R2$:FILES
   :PRINT @282,AA;:PRINT @286,"Filename?
   (or ENTER)";:E=1:F=1:G=12:H=307:A=" "
   :GOSUB 102:IF A="Menu " THEN MENU
138 F$=A:IF F$=" " THEN 141 ELSE OPEN F$ FOR
   INPUT AS 1:INPUT #1,MT:PRINT @282,AA;
   :PRINT @290,"Please Wait...";
139 FOR Q=1 TO 199:P(Q)=Q-1:N(Q)=Q+1:LINE
   INPUT #1,A(Q):IF ASC(A(Q))=LU+1 THEN
   LU=LU+1
140 IF A(Q)<>"end" THEN NEXT ELSE CLOSE
   :A(Q)=CHR$(0)+A(Q):P(1)=Q:N(Q)=1:QT=Q-1
   :IT=Q+1:RETURN
141 A(1)=CHR$(1)+"Outline":P(1)=2:N(1)=2
   :A(2)=CHR$(0)+"end":P(2)=1:N(2)=1:IT=3
   :QT=1:LU=1:RETURN
142 IF PEEK(65442)=1 THEN MENU ELSE X=1
   :CLS:PRINT "Saving your thoughts";:IF
   F$=" " THEN INPUT "to what";F$
143 CLOSE:OPEN F$ FOR OUTPUT AS 1:PRINT
   #1,MT:FOR Y=1 TO QT+1
144 PRINT #1,A(X):X=N(X):NEXT:CLOSE
145 MENU
146 M=L:IF MT=0 THEN MT=-1:LT=L ELSE MT=0
   :LT=1
147 GOSUB 13:GOSUB 48:GOSUB 132:RETURN
150 C$="z!="+MID$(A,E)+CHR$(0):C!=VARPTR(C$)
   :C!=PEEK(C!+1)+PEEK(C!+2)*256
151 CALL 1606,0,C!:CALL2499,0,63105
152 A=LEFT$(A,E-1)+STR$(Z!)+" ":PRINT
   @H+F+E,ER$;:E=E-1:RETURN
155 SOUND1E4,20:IF ERR=52 OR ERR=55 OR
   ERR=54 OR (ERR=5 AND (ERL=139 OR
   ERL=138)) THEN AB="Sorry, thats not a
   THINK.IT file":CLOSE:RESUME 137
157 IF ERL=151 THEN RESUME102
158 CLS:PRINT ERR "in line" ERL:END

```

COOPERATIVE COMPUTER

Menu Design

- * Are menus used extensively? Remembering a lengthy list of command options is hard for the occasional user -- so simplify the process with a short menu of essential choices.
- * If there are many menus, are there shortcuts? Can experienced users issue direct commands or specify a complete menu path at the beginning of the program?
- * Are menus simple and logical? Try to keep each list short. Use multiple levels when the menu becomes too complex to be read at a glance. Try capitalizing the keywords in lengthy choice descriptions.
- * Does each menu have a unique name or number? Verbal and written instructions can then direct users to particular menus as needed.
- * Is it easy to choose a menu option? Pressing a single key is faster than hitting Return after the selection, especially when there are many layers of menus.
- * Are obvious keys used for the selection? Alphabetic keys are best when the choices begin with various letters: A for Add, D for Delete, P for Print. Numbering these options or using PF keys requires a mental table-lookup process -- most typists can find the alpha keys faster.
- * Are menus consistent throughout the program and other software? Many programmers develop an individual style of menu design -- and users appreciate the consistency. If ESC always means, "escape to previous menu," each package will be easier to learn.

Keep It Moving

- * Are long pauses kept interesting? Studies have shown that pauses of more than five seconds appear interminable. That's why telephone hold devices play music.
- * Does the user know when a pause isn't an error? Warn the user: Display the message "Working...." in the middle of the screen for pauses of between five and thirty seconds.
- * Is the user involved? It's little work to inform the audience that the current operation is reading... sorting... writing... verifying... ready. Whenever possible, give progress reports: Read 10 records, 20 records, 30 records.
- * Is the length of the pause known? If a given operation's real time is constant, say, "Be back in three minutes." If the speed of file input or output is calculable, tell the user, "Reading and processing 103 records, estimated time 4 minutes."

Be Considerate

- * Don't use many audible tones. The Tandy is designed to be used in public -- and beeps in a busy office are embarrassing. If you like to include sounds, offer a chance to disable the tones at the start of the program.
- * Before opening work files for output, test to see if a file with that name already exists. If it does, pick another name from an internal "temp file" list or ask the user: "May I erase TEMP.DO? <Y>es, <N>o and return to the menu."
- * Store the date and time with each time the data file is modified. A display of "File last accessed 7/4/76" can warn users of potential problems.
- * When asking for the name of a data file, display the disk directory. It's not easy to remember all those names!

- * If the program writes to a physical disk, check for free space and write protection early in the program. If necessary, alert the user: "Only 15K free space. Do you wish to <C>hange disks, <P>roceed or <Q>uit?"
 - * If the application modifies the original data file, create a backup. If this isn't possible, ask the user: "Have you backed up INPUT.DO? <Y>es, proceed <N>o, quit application."
 - * Test for the data file's presence before asking for program options. It's annoying to type lines of application specifications, only to see "File Not Found". If the file's not found, offer the user the chance of correcting the name: "File not found, please re-enter name. Press Return to quit."
 - * If the application requires standard data or routine files, check to see if they're all present. The earlier this test is made, the better.
 - * At the start of a program, display the name of the program with a prompt allowing the operator to quit. It's amazing how many times users select -- and run -- the wrong application.
 - * Offer a brief description of the program at the above prompt. If the user's not sure what the program does, offer a reminder: "Weekly Calendar Program. <P>roceed, <Q>uit or <H>elp?"
 - * Prompt the user to connect and activate the printer before attempting to output to the device. Also advise the user to turn off the printer -- it might be battery-powered. This applies to other peripherals as well, such as the disk drive.
 - * Include a quick up and running section at the start of the manual. Most owners can't wait to use their new program -- help them along. List any hazards at the beginning as well -- don't bury warnings in the body of the manual.
 - * If the documentation is more than a few pages, make a comprehensive index. Don't assume that the table of contents or quick-reference guide can replace the index.
 - * Basic machine information can be placed into an appendix. Advanced users won't need a guide to making back-up copies of a cassette, so guide novices to the back of the manual.
 - * Use diagrams as often as possible. Show all important screens, instead of attempting to describe them verbally.
 - * Number parts of a difficult procedure, such as installing the program or producing a complex report. Use statements like: "If you're using a dot-matrix printer, go to step 15."
 - * Interesting but irrelevant technical information doesn't belong in a set of directions. Refer users to a technical appendix or a later chapter.
 - * If the program code can be customized by the user, include a comprehensive variable list in the documentation along with a block diagram of the listing.
 - * If the program uses specific mathematical formulas, describe these in the manual. If government or other standard sources were used, refer to these in a bibliography.
- Documentation**

ONE-LINERS FOR THE MASSES

An upcoming issue of PPR shall be devoted to mass-storage devices for the Model 100 and Tandy 200. These devices currently include several 5.25-inch and 3.5-inch disk drives, bubble memories and various cassette-based media.

What we'd like: a collection of short tips and utilities for using mass storage devices with laptop computers.

We're not going to pay, of course, for a one or two-line suggestion about cleaning drive heads or running the Disk/Video interface from an automobile battery.

But we'll give full credit where credit is due. So, tell us your one-liners. And send us that neat routine that makes your mass-storage devices a little easier to deal with.

Of course, if you have a suggestion for making programs friendlier or faster (see PPR, August '85), tell us about that, too.

Thanks, and we're looking forward to your cards and letters.

Alan L. Zeichick
Editor

PORTABLE PROGRAM REVIEW

P.O. Box 250
Highland Mill
Camden, ME 04843

FIRST CLASS U.S. POSTAGE PAID CAMDEN, ME PERMIT NO. 5

CUSTOMER: PLEASE ADVISE IF ABOVE ADDRESS IS INCORRECT